Virtual Laboratories for Training in Industrial Robotics

J. A. Ortega, R. E. Sánchez, J. J. González and G. Reyes

Abstract— **This paper presents a methodology for the development of virtual laboratories for training in industrial robotics. The methodology starts with the determination of the technical specifications of the equipment and processes to be virtualized, besides the specific didactic requirements to be implemented within each scenario. The simulation stage includes modeling the dynamic behavior of real physical systems, such as sensors and actuators, which allows the user to not only observe basic concepts related to robotics, but can also develop more practical skills, for example, the instrumentation and control of a robotic systems. The virtualization of a Delta-type robot is presented as a case study to show the application of the proposed methodology.**

Keywords— **Industrial robotics, Modeling of dynamics systems, Real time simulation, Virtual laboratory.**

I. INTRODUCCION

LA PRESENCIA de sistemas robóticos es cada vez más frecuente en la industria manufacturera alrededor del mundo, lo cual hace un llamado directo a las instituciones educativas y centros de capacitación hacia la implementación de estrategias más eficientes para el desarrollo de habilidades prácticas en sus estudiantes relacionadas concretamente con el diseño, implementación y operación de sistemas robóticos.

Son muy conocidos los inconvenientes a los que se enfrentan las escuelas y centros de capacitación para dotarse y mantener un laboratorio completo de robótica industrial: los robots son caros, requieren de mantenimiento, los estudiantes pueden dañar involuntariamente algunos componentes del sistema, etc. Por otro lado, el hecho de tener un sistema robótico en el laboratorio no garantiza el adecuado aprendizaje en los estudiantes, pues muchas veces estos sistemas son cerrados, lo cual impide que se tenga una experiencia más enriquecedora que la de simplemente manipular el robot a través de una interfaz gráfica. Es precisamente ahí donde se empiezan a valorar las bondades que las herramientas de simulación, particularmente la realidad virtual, ofrecen para sobrellevar este tipo de inconvenientes.

J. A. Ortega, Instituto Tecnológico Superior de San Andrés Tuxtla. Carretera Costera del Golfo S/N, Km. 140+100, C.P. 95804. San Andrés Tuxtla, Veracruz, México. (email: jorgemoody@gmail.com).

R. E. Sánchez, Instituto Politécnico Nacional. CICATA-Qro. Cerro Blanco No. 141 Col. Colinas del Cimatario, C.P. 76090. Querétaro, Querétaro, México. (email: rogersan1984@hotmail.es).

J. J. González, Instituto Politécnico Nacional. CICATA-Qro. Cerro Blanco No. 141 Col. Colinas del Cimatario, C.P. 76090. Querétaro, Querétaro, México. (email: jgonzalezba@ipn.mx).

G. Reyes, Instituto Tecnológico Superior de San Andrés Tuxtla. Carretera Costera del Golfo S/N, Km. 140+100, C.P. 95804. San Andrés Tuxtla, Veracruz, México. (email: greyesm_13@hotmail.com).

Actualmente las plataformas para desarrollos virtuales, motores de física, procesadores de cómputo y procesadores gráficos, han contribuido con el surgimiento de aplicaciones virtuales cuyo propósito, a diferencia de aplicaciones convencionales, va más allá del entretenimiento. Esta nueva tendencia conocida en inglés como "Serious Games" tiene su principal campo de acción en la instrucción y el entrenamiento [1], lo cual se puede evidenciar en diferentes contextos, por ejemplo en la industria [2, 3], la salud [4-8], la educación [9-15], lo militar [16], la seguridad pública [17-18], etc.

Muchos de los conceptos de los "Serious Games" han sido implementados para el desarrollo de aplicaciones virtuales que simulan escenarios de robótica industrial con fines educativos [19-22], incluso ya se encuentran disponibles programas computacionales comerciales que ofrecen una experiencia visual altamente realista para la interacción usuario-robot [23]. A pesar de lo anterior, los alcances de estas aplicaciones con fines realmente prácticos para el proceso de entrenamiento son algo limitados. Estos alcances se enfocan en algunos conceptos de la robótica, por ejemplo la movilidad y la cinemática del mecanismo, y la descripción de rutas a seguir dentro del espacio de trabajo del robot, olvidando uno de los aspectos fundamentales; la instrumentación y el control. Existen algunas plataformas educativas que sí consideran este tipo de conceptos [24-27], sin embargo, se concentran exclusivamente en la etapa de virtualización de algoritmos de control para sistemas robóticos reales, por lo que se puede decir que no son laboratorios completamente virtuales.

En las aplicaciones convencionales los robots son animados considerando algunos aspectos físicos como la gravedad, la interacción entre cuerpos, la velocidad de los movimientos, etc., con lo cual se logra una simulación de alta calidad visual. Pese a lo anterior, debido a que ninguno de los elementos del sistema robótico es modelado como un sistema dinámico, la aplicación de una etapa virtual de control resulta técnicamente imposible. Lo anterior no sólo representa una limitante didáctica para el aprendizaje práctico de la robótica, también representa un problema para la simulación misma, pues el comportamiento obtenido en el robot al moverse no representa de forma fiel el comportamiento de un robot real.

Tomando en cuenta lo anterior es que en este trabajo se presenta una metodología para el diseño e implementación de laboratorios virtuales de robótica industrial. La idea central es que estos laboratorios incluyan las características físicas elementales de los componentes que integran a los robots y al ambiente que los rodea, así como el modelado de elementos que permitan aumentar la experiencia de operación real de un robot industrial, como sensores, actuadores y algoritmos de

control.

II. Metodología Propuesta

La metodología propuesta (ver Fig. 1) está compuesta por las etapas que se describen a continuación:

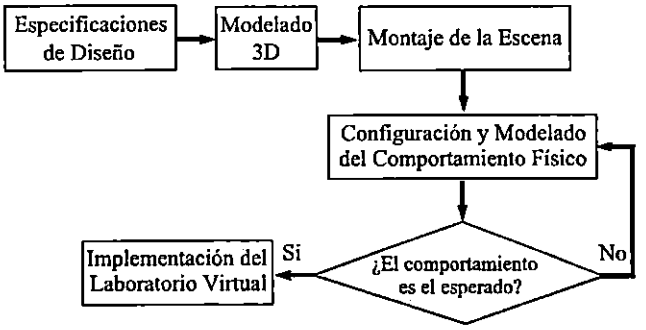

Figura 1. Metodología propuesta.

A. Determinación de las Especificaciones de Diseño

En esta etapa se puede distinguir dos tipos de especificaciones de diseño; las especificaciones técnicas y las especificaciones didácticas del laboratorio. Las especificaciones técnicas van desde el tipo de proceso y layout a implementar, hasta la determinación de las características particulares de los componentes y equipos a simular. Por otro lado, en las especificaciones didácticas se define el alcance del laboratorio dentro del proceso de capacitación, lo cual está relacionado con la determinación de los conceptos a estudiar, el tipo de habilidades a desarrollar y las formas específicas a implementar para el desarrollo de estas habilidades.

B. Modelado 3D de los Componentes del Laboratorio

Gran parte de la experiencia realista de un ambiente virtual depende del esfuerzo que se concentre en la elaboración de los modelos 3D presentes en la escena. En la actualidad se dispone de muchas herramientas de diseño CAD y modelado 3D que permiten obtener diseños de mecanismos robóticos con un elevado nivel de detalle. Sin embargo, hay que tener presente que mientras mayor sea el nivel de detalle en la escena, mayor será el costo computacional de la aplicación virtual desarrollada, por lo que en esta etapa se debe tener cuidado en modelar a detalle sólo aquellos componentes estrictamente necesarios.

C. Montaje de la Escena Dentro de la Plataforma Virtual

Todos los modelos 3D obtenidos en la etapa anterior son insertados dentro de una plataforma de desarrollo virtual para ir formando el layout del laboratorio. En esta etapa se debe hacer especial énfasis en el ensamblaje de los robots dentro de la escena, por lo que el nivel de complejidad en esta tarea depende del nivel de detalle en el diseño de sus componentes.

D. Configuración y Modelado del Comportamiento Físico

La metodología propuesta está diseñada para implementarse sobre plataformas de desarrollo virtual que incluyan motores de física de alto desempeño, lo cual permite representar las condiciones y características físicas elementales para poder simular la dinámica de cuerpos rígidos. En este sentido los desarrolladores de este tipo de aplicaciones deben tener un alto grado de conocimiento de las características que su plataforma virtual les permite implementar o configurar. Básicamente se requiere una plataforma que permita simular la fuerza gravitatoria, la detección de colisiones, las articulaciones básicas utilizadas en robótica, las propiedades inerciales de los cuerpos, fuerzas, torques, etc.

La plataforma debe incluir además la posibilidad de desarrollar código de programación que permita agregar al entorno virtual la representación matemática de todos los sistemas dinámicos involucrados y su etapa de control. Para la aplicación objeto de estudio en este trabajo los sistemas dinámicos que vale la pena modelar son los servomotores de los robots, los cuales son básicamente sistemas electromecánicos que incluyen un motor de corriente directa, un sensor de posición (encoder) y una etapa de control PID. Lo anterior no significa que la metodología se limita al modelado de este tipo de actuadores, sensores y algoritmos de control, por el contrario, la metodología es tan abierta que se puede incluir dentro del laboratorio cualquier otro tipo de sistema que se pueda controlar y que pueda interactuar con un sistema robótico, por ejemplo sistemas térmicos o hidráulicos, los cuales son sistemas cuyo modelado ya ha sido muy estudiado. Esta etapa de modelado concluye hasta que se obtiene el comportamiento físico esperado en todos los sistemas embebidos en el escenario. Para mayor información acerca del modelado y control de sistemas dinámicos revisar [28-29].

E. Implementación del Laboratorio Virtual

La implementación consiste en armonizar todas las etapas mencionadas anteriormente para obtener una aplicación virtual que represente un laboratorio de robótica industrial. El propósito es que el usuario pueda observar el comportamiento real de un sistema robótico y tenga a su disposición un conjunto de herramientas, como indicadores y controles, que le permitan interactuar con el sistema y de esta forma desarrollar habilidades prácticas de la robótica.

Las aplicaciones virtuales derivadas de la metodología expuesta anteriormente pueden ser vistas como arquitecturas desarrolladas en tres niveles (ver Fig. 2). El primer nivel, llamado "Ambiente 3D", se desarrolla en el área gráfica de la plataforma virtual, y es ahí donde el motor de física actúa durante la simulación para representar en tiempo real el comportamiento de todos los cuerpos rígidos en movimiento, de las articulaciones que unen dichos cuerpos rígidos y de las condiciones ambientales que están inmersas dentro del escenario. El segundo nivel, llamado "Instrumentación y Control", se desarrolla mediante código de programación. En este nivel básicamente se emulan los actuadores, sensores y algoritmos de control que permiten representar el movimiento automatizado de un mecanismo, diferenciándose así la aplicación virtual desarrollada de una simple animación 3D. Dependiendo del propósito y alcance del laboratorio virtual a

implementar, algunos sensores y actuadores pueden ser virtualizados de forma muy simple, por ejemplo sensores de proximidad y efectores finales, sin embargo otros dispositivos, cuya participación es crucial para la simulación y para la consecución de los objetivos de entrenamiento, requieren ser modelados como sistemas dinámicos, por ejemplo motores eléctricos, pistones hidráulicos, etc. En este nivel también se pueden programar secuencias a realizar por el robot o por las celdas robóticas presentes en el escenario. El tercer nivel, llamado "HMI" por las siglas en inglés de Human Machine Interface, se desarrolla mediante código de programación, pero su despliegue durante la simulación se realiza sobre el área gráfica. A través de esta interfaz el usuario puede configurar en tiempo real algunas características de interés tanto del nivel "Ambiente 3D" como del nivel "Instrumentación y Control", lo cual representa una herramienta de gran alcance para el desarrollo de algunas prácticas específicas. Por ejemplo el usuario puede: configurar la masa de cada elemento del robot para evaluar la capacidad de carga de un actuador, modificar las características mecánicas y eléctricas de un motor para apreciar su comportamiento, sintonizar el controlador implementado, visualizar el status del sistema, visualizar señales de control, configurar rutinas de trabajo y operar el robot.

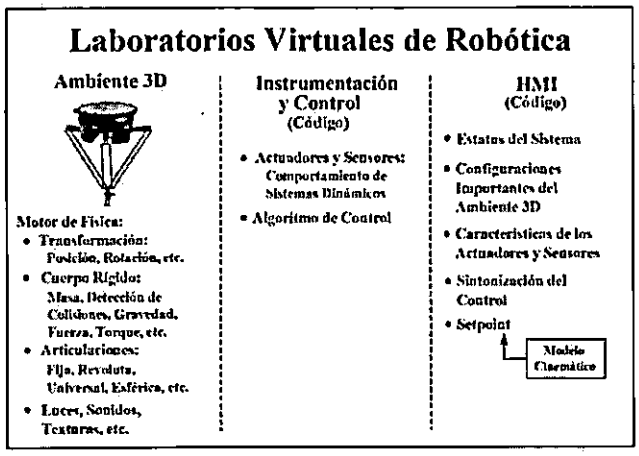

Figura 2. Arquitectura de los laboratorios virtuales.

Desde finales de los años 90's se introdujo el concepto de "Harware in the loop" [30] como una técnica de simulación en tiempo real que se basa en el modelado matemático de sistemas complejos, sin embargo, debido a las limitantes tecnológicas de aquella época, este concepto no estaba disponible para todos los niveles de la comunidad académica. En la actualidad esta situación es diferente, hoy cualquier persona puede tener acceso a computadoras con capacidad de cálculo y despliegue de gráficos suficiente para poder aplicar este tipo de conceptos, y es ahí donde recae el principal aporte de la metodología propuesta en este trabajo.

III. Caso de Estudio: Modelado y Virtualización de un Robot Tipo Delta

Para este caso de estudio la plataforma de desarrollo virtual utilizada es Unity 3D en su versión 4.6, la cual cuenta con el motor de física PhysX™. Esta plataforma permite desarrollar código de programación en lenguajes de alto nivel como C# y Java, los cuales pueden ser elaborados en IDEs como Microsoft Visual Studio o MonoDevelop (Licencia Pública General). A continuación se presenta el desarrollo de este caso de estudio.

A. Especificaciones de Diseño

El robot a implementar es un manipulador paralelo tipo Delta de 3 grados de libertad de traslación pura. La aplicación a desarrollar debe incluir las características físicas elementales para poder representar el mecanismo, es decir el robot debe estar sujeto a la acción de la gravedad, sus componentes deben incluir todas propiedades inerciales de un cuerpo rígido, la movilidad de las articulaciones activas y pasivas debe ser simulada y cada componente debe ocupar un único espacio (detección de colisiones). Se incluirá una cámara virtual para visualizar en tiempo real a través de una pantalla el movimiento del efector final del robot. El robot debe estar habilitado para manipular objetos dentro de su espacio de trabajo, para ello se debe simular un efector final y un sensor de proximidad. Los servomotores del manipulador deben estar modelados como sistemas dinámicos y una etapa de control PID será implementada para controlar su posición angular. La movilidad del manipulador debe estar basada en su modelo cinemático inverso. Finalmente, se agregarán elementos extras como objetos no animados, colores y luces para dar un acabado profesional al escenario virtual.

Con respecto al uso de la aplicación y el desarrollo de prácticas de laboratorio, se desarrollará una interfaz virtual que cuente con una serie de menús que permitan: (i) revisar el estatus general de los sensores y actuadores del robot, (ii) configurar la masa de los componentes principales del robot, (iii) configurar las propiedades eléctricas y mecánicas de los motores, (iv) sintonizar el controlador PID, deshabilitar la acción de control y controlar en lazo abierto cada actuador, y (v) operar el robot obteniendo los SetPoints de sus actuadores a través de su modelo cinemático inverso.

B. Modelado 3D

Todos los modelos CAD utilizados fueron desarrollados a escala real en SolidWorks®, luego fueron exportados en formato STL (Stereo Lithography) hacia 3DS MAX, en donde se trabajó en la reducción de los polígonos que forman el mallado de cada modelo 3D. El propósito de este tratamiento es disminuir el costo computacional del despliegue gráfico en la aplicación virtual. Posteriormente cada modelo 3D es enviado en formato FBX (FilmBox) hacia Unity 3D.

C. Montaje y Configuración del Comportamiento Físico

Los componentes que forman parte de la escena se pueden dividir en tres grupos; objetos no animados, objetos animados y objetos cuyo comportamiento físico debe ser modelado. Los objetos no animados y animados son esos elementos extras que decoran el ambiente y dan una vista profesional del entorno, la diferencia entre ellos es que unos se mueven y otros no. El montaje de este tipo de componentes no es de

gran importancia, sin embargo en el caso de los robots, cuya movilidad debe ser modelada, se debe hacer mucho hincapié, pues errores en el ensamblaje del robot generarán errores en el posicionamiento preciso de su efector final, tal y como sucede con un robot real.

El robot Delta presenta una estructura basada en tres cadenas cinemáticas idénticas unidas en sus extremos distales por una plataforma móvil. En la Fig. 3 se muestra el montaje de una de las tres cadenas cinemáticas del robot dentro de Unity 3D. En dicha figura se observa un brazo (con forma prácticamente cilíndrica), un antebrazo (integrado por dos cuerpos formando un paralelogramo) y la plataforma móvil (de forma triangular). Cada uno de estos componentes debe ser configurado dentro de Unity 3D con la opción "Rigid Body" (Cuerpo Rígido), de otra manera sobre este cuerpo no podrá tener efecto el motor de física, lo cual significa que no actuará la gravedad, no se le podrá asignar masa, un material específico, detección de colisiones, aplicar una fuerza, un torque, etc.

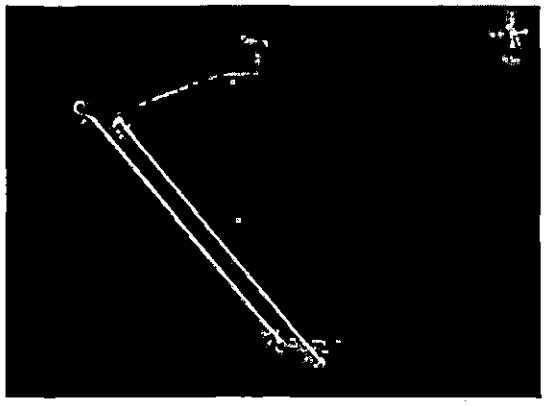

Figura 3. Montaje de una cadena cinemática del robot dentro de Unity 3D.

Para convertir el montaje mostrado en la Fig. 3 en un sub-ensamble del robot se deben simular las articulaciones que conectan a cada cuerpo con otro. El arreglo articular de cada cadena cinemática del robot es el siguiente; el brazo está conectado a tierra (plataforma fija) mediante una articulación de revoluta, cada cuerpo que forma el antebrazo está conectado al brazo mediante una articulación universal, y finalmente la plataforma móvil está unida a cada cuerpo del antebrazo también mediante articulaciones universales. En Unity 3D se puede simular la movilidad y el comportamiento de prácticamente cualquier articulación comúnmente utilizada en robótica; articulaciones de revoluta, prismáticas, universales, esféricas, etc., y gran parte de ello se debe a la disponibilidad de la herramienta "Configurable Joint" (Articulación Configurable). En la Fig. 4 se muestra una parte del menú de opciones para la configuración de una articulación que conecta uno de los cuerpos de un antebrazo con la plataforma móvil (llamada Mplat) en el punto X = -0.0047 m, Y = -0.31196 m, Z = 0.2526 m de su sistema de referencia local. Dicha articulación representa una de tipo universal, pues tal y como se observa el movimiento lineal sobre los ejes X, Y y Z, además del movimiento angular sobre el eje Y fueron convenientemente bloqueados.

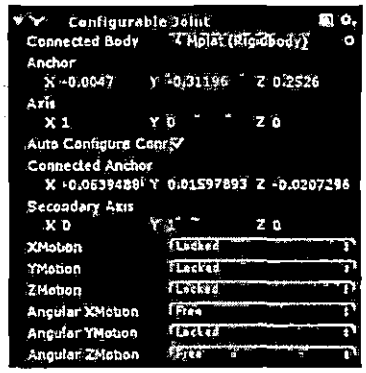

Figura 4. Ejemplo de la configuración de una articulación universal en Unity 3D.

Sobre cada cuerpo del ensamblaje se debe configurar un "Collider" (Colisionador) para simular su interacción con otros cuerpos dentro del ambiente. Un collider es una región envolvente que delimita el espacio de contacto de un cuerpo. Los tipos de collider disponibles en Unity 3D son: Box Collider (tipo caja), Sphere Collider (tipo esfera), Capsule Collider (tipo cápsula) y Mesh Collider (tipo malla). En la Fig. 5 se muestra la aplicación de un Capsule Collider a uno de los brazos del robot Delta.

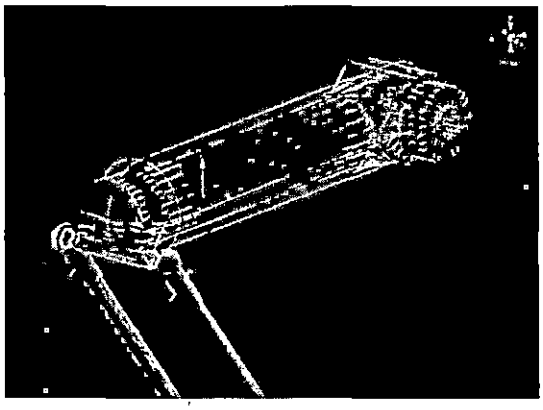

Figura 5. Aplicación de un Capsule Collider a un brazo del robot.

Cuando el procedimiento anterior se aplica a todos los componentes del robot se obtiene su modelo virtual. En la Fig. 6 se muestra el robot completamente ensamblado. En esta figura se observa las cadenas cinemáticas del robot pendiendo de la base fija bajo únicamente la acción de la gravedad.

Finalmente se agregan algunos elementos extras para mejorar el aspecto visual del escenario. En la Fig. 7 se muestra una vista del escenario en donde se puede apreciar la inclusión de luces, colores y otros objetos, entre los que destaca un monitor que proyecta en tiempo real el streaming de video captado por una cámara virtual instalada en el robot, además de la presencia de cinco cubos de colores dispuestos sobre una mesa para ser manipulados por el robot.

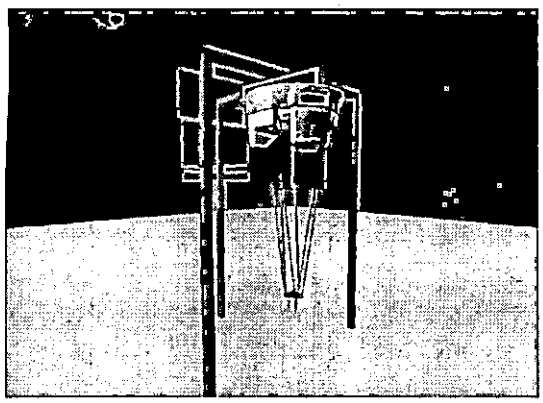

Figura 6. Modelo virtual del robot Delta bajo la acción de la gravedad.

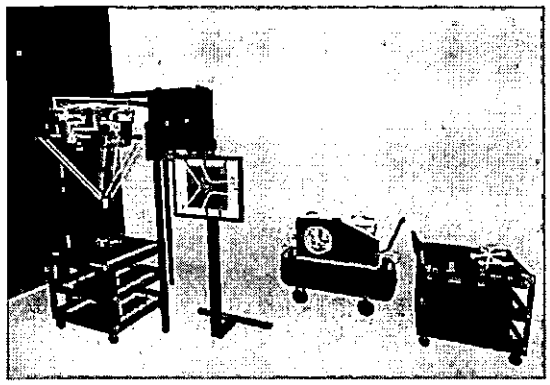

Figura 7. Escenario virtual del robot Delta.

Hasta este punto el modelo virtual desarrollado representa las características de un mecanismo paralelo sin ningún tipo de sensores y actuadores para controlar su movimiento. Para mayor información acerca de la configuración geométrica de un robot Delta revisar [31].

D. Instrumentación y Control Virtual

A continuación se presenta la forma en que se modeló esta etapa para el robot Delta.

Sensor de proximidad y efector final: Ambos dispositivos se pueden simular fácilmente en Unity 3D a través de código de programación muy simple. En el caso del sensor de proximidad se utiliza la opción "Trigger" (Disparo) de los colliders, de manera que cuando el collider del efector final entre en contacto con el collider de otro objeto, una variable booleana previamente definida toma valor verdadero, misma variable que toma valor falso cuando los colliders no están en contacto. El rango de acción del sensor está íntimamente relacionado con la posición y tamaño del collider del efector final. Por otro lado, el efector final es simulado utilizando la opción "Parent" de los objetos definidos dentro de Unity 3D, de esta manera un objeto se puede adjuntar o separar jerárquicamente del efector final del robot mediante la acción de una variable booleana.

Servomotores DC: Estos dispositivos juegan un papel importante dentro del proceso de simulación del robot, por ello deben ser modelados como sistemas dinámicos, es decir conviene no sólo conocer su respuesta en el estado final, también es necesario obtener su comportamiento en el estado transitorio. El modelado de un servomotor de corriente directa implica la representación de básicamente tres sub-sistemas: un motor de corriente directa, un sensor de posición angular y una etapa de control.

En lo que respecta al motor de corriente directa es bien sabido que su representación está basada en las siguientes ecuaciones diferenciales:

$$v_{(t)} = Ri_{(t)} + L\frac{di_{(t)}}{dt} + e_{(t)} \tag{1}$$

$$T_{(t)} = J\frac{d\omega_{(t)}}{dt} + B_m\omega_{(t)} \tag{2}$$

Las ecuaciones (1) y (2) representan respectivamente la distribución del voltaje de alimentación $v_{(t)}$ del motor en sus elementos eléctricos y la distribución del par $T_{(t)}$ en sus elementos mecánicos. Donde; $i_{(t)}$ es la corriente de la armadura, R la resistencia eléctrica, L la inductancia eléctrica, $e_{(t)}$ la fuerza electromotriz, J el momento de inercia del rotor, $\omega_{(t)}$ la velocidad angular del eje, y B_m la constante de fricción viscosa del motor.

Además se sabe que el par generado por el motor es proporcional a la corriente, y que la velocidad angular del eje del motor es proporcional a fuerza electromotriz, lo cual se puede representar de la siguiente forma:

$$T_{(t)} = k_t i_{(t)} \tag{3}$$

$$e_{(t)} = k_m\omega_{(t)} \tag{4}$$

Donde k_t y k_m son respectivamente la constante de par y la constante de voltaje del motor.

De lo anterior, aplicando la transformada de Laplace sobre (1) y (2) se obtiene lo siguiente:

$$V_{(S)} = RI_{(S)} + LSI_{(S)} + k_m\Omega_{(S)} \tag{5}$$

$$k_tI_{(S)} = JS\Omega_{(S)} + B_m\Omega_{(S)} \tag{6}$$

Despejando $I_{(S)}$ de (6) y sustituyéndola en (5) se puede llegar a la siguiente función de transferencia de lazo abierto, en donde la velocidad angular $\Omega_{(S)}$ de la flecha del motor es considerada la salida, mientras el voltaje de alimentación $V_{(S)}$ es considerado la entrada.

$$\frac{\Omega_{(S)}}{V_{(S)}} = \frac{k_t}{(JS + B_m)(LS + R) + k_tk_m} \tag{7}$$

La expresión (7) representa un sistema continuo en el dominio de Laplace, y para poder implementarla como un sistema discreto en Unity 3D, se debe aplicar la transformada

bilineal (método Tustin), la cual se resume de la siguiente forma:

$$S \leftrightarrow \frac{2}{T}\frac{Z-1}{Z+1} \tag{8}$$

donde T representa el período de muestreo, en el caso particular de Unity 3D $T = 0.02$ s.

De lo anterior se puede calcular la velocidad angular de la flecha de un motor con determinadas características eléctricas y mecánicas dado un voltaje de entrada, lo cual junto con el hecho que las articulaciones simuladas en Unity 3D pueden ser configuradas como activas e inducirles dicha velocidad angular, permite simular en tiempo real el movimiento de un eslabón activo generado por un motor de corriente directa.

Por otro lado, el sensor de posición del servomotor se puede simular a través de la lectura en tiempo real de la posición angular del eslabón activo sobre su eje de rotación. La diferencia entre un SetPoint de posicionamiento angular (SP) y el valor actual (PV) registrado por el sensor de posición virtual permite conocer el error $E_{(n)}$ de posicionamiento en un instante n del tiempo, el cual es un dato muy importante para la implementación de la etapa de control del servomotor.

La etapa de control de posición implementada en los servomotores es de tipo PID, en donde la señal de control V_{in} se estima por la sumatoria de una acción de control proporcional (P), una integral (I) y una derivativa (D). Estas acciones de control fueron definidas en función del error $E_{(n)}$ tal y como se muestra a continuación:

$$P_{(n)} = K_p E_{(n)} \tag{9}$$

$$I_{(n)} = K_i T\left(E_{(n)} + I_{(n-1)}\right) \tag{10}$$

$$D_{(n)} = K_d\left(\frac{E_{(n)} + E_{(n-1)}}{T}\right) \tag{11}$$

Donde K_p, K_i y K_d representan las ganancia proporcional, integral y derivativa que permiten sintonizar el controlador para obtener un comportamiento deseado.

E. Desarrollo de la Interfaz de Usuario

La idea principal de esta interfaz es que el usuario pueda interactuar de forma activa con el robot y sus principales opciones de configuración. Para esta aplicación se desarrolló una interfaz con cinco menús los cuales se pueden observar en la Fig. 8 y cuya funcionalidad se describe a continuación:

Menú Status: la función principal de este menú es mostrar el estado de algunas variables durante la operación del robot. Se puede leer la posición angular actual de cada motor y su máxima velocidad angular. En la parte inferior se encuentran tres indicadores: "Position Reached", el cual indica cuando los motores han llegado a su SetPoint, "Efector Sensor", que indica cuando la pieza que deseamos manipular ya está dentro del campo de acción del efector final, y "Efector On/Off", que indica cuando el efector final fue activado o desactivado para manipular una pieza.

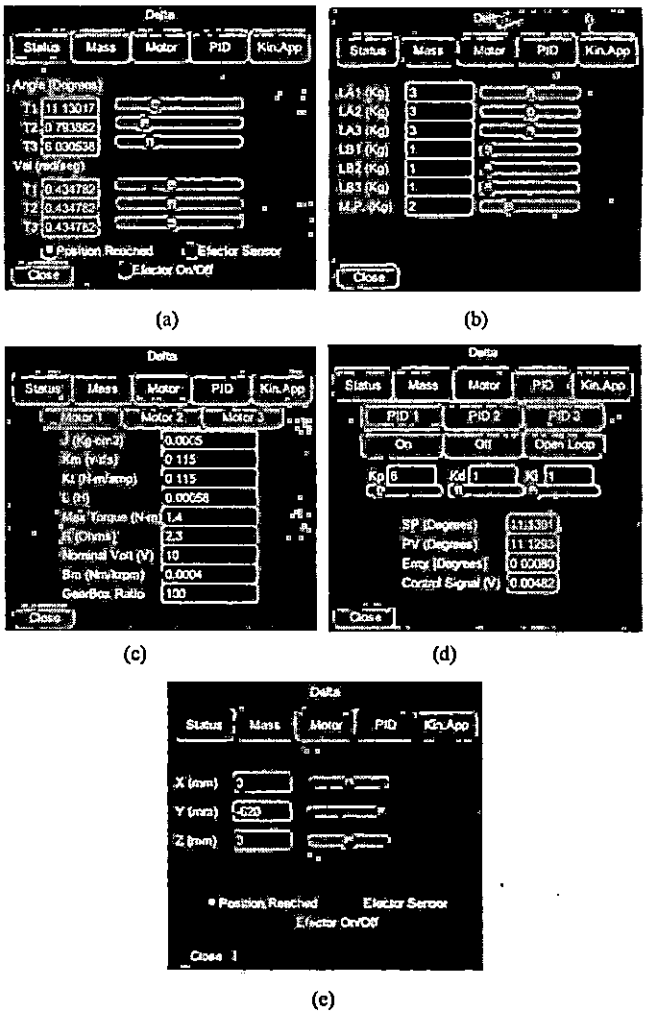

Fig. 8. Interfaz de usuario. (a) Menú Status. (b) Menú Mass. (c) Menú Motor. (d) Menú PID. (e) Menú Kin. App.

Menú Mass: Permite configurar en tiempo real la masa de los eslabones del robot. Tomando en cuenta las $i = 1, 2, 3$ cadenas cinemáticas del robot, la etiqueta para cada brazo es LA_i, para cada antebrazo LB_i, y para la plataforma móvil $M.P.$ La asignación de la masa para los antebrazos se divide entre los dos cuerpos que lo conforman. La simulación alcanza un nivel de realismo talque si se seleccionan masas que no pueden ser cargadas por los motores se verá cómo el robot o las cadenas cinemáticas con menos capacidad de carga no pueden sostenerse.

Menú Motor: Este menú permite configurar las características eléctricas y mecánicas de cada uno de los motores del robot. En ese menú aparecen no sólo las características mostradas en la sección de modelado de un motor de corriente directa, también se incluyó el Máximo torque y la relación de reducción en los engranajes del servomotor. La asignación del Máximo torque es posible gracias a una de las opciones avanzadas de la "Configurable Joint" cuando es configurada como activa.

Menú PID: Este menú permite configurar algunos parámetros relacionados con el lazo de control PID de cada motor. Las primeras opciones que nos provee este menú es

seleccionar un modo de operación, en este sentido se puede seleccionar entre las opciones de control: "On", "Off" y "Open Loop". La opción "Off" genera una señal de control igual a cero, lo cual implica básicamente apagar ese motor. La opción "Open Loop" permite indicar manualmente la señal de control, lo cual permite conocer la respuesta natural de cada motor. Finalmente la opción "On" habilita el lazo de control PID, para lo cual está disponible la sintonización de las ganancias del controlador y la visualización del SetPoint, el valor actual del proceso, el error y la señal de control.

Menú Kin. App: Permite operar el robot, lo cual implica embeber en la aplicación el modelo cinemático inverso del robot [31] para encontrar así las coordenadas articulares de cada actuador asociadas a cualquier punto o trayectoria en su espacio de trabajo. El usuario puede realizar movimientos en los ejes *X*, *Y* y *Z*, y tiene disponibles los indicadores de "Position Reached" y "Efector Sensor", además de un control para activar y desactivar el efector final "Efector On/Off". En la Fig. 9 se muestra el despliegue de este menú para operar el robot y manipular los objetos dentro de su espacio de trabajo.

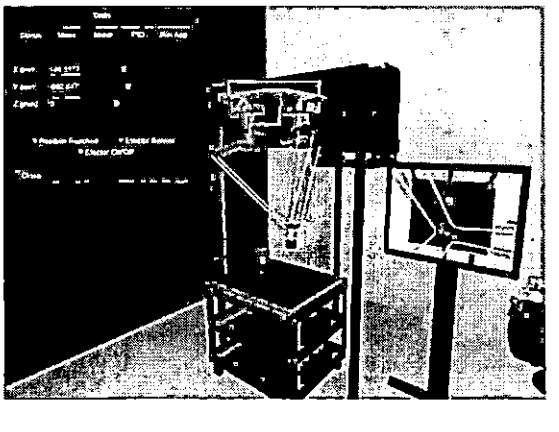

Figura 9. Vista del robot Delta apilando objetos en su espacio de trabajo.

IV. CONCLUSIONES

Se presentó una metodología sistemática para el desarrollo de laboratorios virtuales con propósitos de entrenamiento en robótica industrial. La metodología se fundamenta en la caracterización de las propiedades físicas de todos los elementos embebidos en el escenario, desde la estructura mecánica de los sistemas robóticos, hasta el modelado dinámico de los sensores y actuadores utilizados para el control y operación del robot.

Este tipo de laboratorios constituyen herramientas de gran utilidad para la capacitación de personal en tópicos prácticos de la robótica, pues el usuario de estos laboratorios puede entre otras cosas: familiarizarse con la operación y control de cualquier robot industrial, visualizar variables importantes del sistema, probar el comportamiento de diferentes actuadores al introducir sus especificaciones técnicas reales, realizar pruebas de control avanzadas como modificar la masa de los eslabones para estudiar el comportamiento del controlador, desarrollar habilidades de programación en lenguajes de alto desempeño, estudiar a detalle e implementar el modelo cinemático de robots industriales, implementar algoritmos de control para planificación y seguimiento de trayectorias, e implementar rutinas y secuencias de trabajo.

REFERENCES

[1] M. Zyda. (2005, Sept.). From visual simulation to virtual reality to games. Computer, 38(9), pp. 25-32.

[2] I.S. Brasil, F.M.M. Neto, J.F.S. Chagas, R. Monteiro, D.F.L. Souza, M.F. Bonates and A. Dantas, "An intelligent and persistent browser-based game for oil drilling operators training," in Serious Games and Applications for Health (SeGAH), 2011 IEEE 1st International Conference on, Braga, 2011, pp. 1-9.

[3] H. Guo, H. Li, G. Chan and M. Skitmore. (2012, Sept.). Using game technologies to improve the safety of construction plant operations. Accident Analysis & Prevention, 48, pp. 204-213.

[4] K. de O Andrade, G. Fernandes, J. Martins, V.C. Roma, R.C. Joaquim and G.A.P. Caurin, "Rehabilitation robotics and serious games: An initial architecture for simultaneous players," in Biosignals and Biorobotics Conference (BRC), 2013 ISSNIP, Rio de Janerio, 2013, pp. 1-6.

[5] R.S. Torres and F.L.S Nunes, "Applying Entertaining Aspects of Serious Game in Medical Training: Systematic Review and Implementation," in Virtual Reality (SVR), 2011 XIII Symposium on, Uberlandia, 2011, pp. 18-27.

[6] O. Erazo, J.A. Pino, R. Pino and C. Fernandez, "Magic Mirror for Neurorehabilitation of People with Upper Limb Dysfunction Using Kinect," in System Sciences (HICSS), 2014 47th Hawaii International Conference on, Waikoloa, HI, 2014, pp. 2607-2615.

[7] R.J. Lancaster. (2014, Mar.). Serious Game Simulation as a Teaching Strategy in Pharmacology. Clinical Simulation in Nursing, 10(3), pp. 129-137.

[8] A. Sliney, "JDoc: A Serious Game for Medical Learning," in Advances in Computer-Human Interaction, 2008 First International Conference on, Sainte Luce, 2008, pp. 131-136.

[9] G.A.E. Khayat, T.F. Mabrouk and A.S. Elmaghraby, "Intelligent serious games system for children with learning disabilities," in Computer Games (CGAMES), 2012 17th International Conference on, Louisville, KY, 2012, pp. 30-34.

[10] A.M. Hussaan and K. Sehaba, "Adaptive Serious Game for Rehabilitation of Persons with Cognitive Disabilities," in Advanced Learning Technologies (ICALT), 2013 IEEE 13th International Conference on, Beijing, 2013, pp. 65-69.

[11] D. Wassila and B. Tahar, "Using serious game to simplify algorithm learning," in Education and e-Learning Innovations (ICEELI), 2012 International Conference on, Sousse, 2012, pp. 1-5.

[12] N. Adamo-Villani, T. Haley-Hermiz and R. Cutler, "Using a Serious Game Approach to Teach 'Operator Precedence' to Introductory Programming Students," in Information Visualisation (IV), 2013 17th International Conference, London, 2013, pp. 523-526.

[13] A.G. Abulrub, A.N. Attridge and M.A. Williams, "Virtual reality in engineering education: The future of creative learning," in Global Engineering Education Conference (EDUCON), 2011 IEEE, Amman, 2011, pp. 751-757.

[14] A. Schäfer, J. Holz, T. Leonhardt, U. Schroeder, P. Brauner and M. Ziefle. (2013, Apr.). From boring to scoring - a collaborative serious game for learning and practicing mathematical logic for computer science education. Computer Science Education, 23(2), pp. 87-111.

[15] S. Arnab, K. Brown, S. Clarke, I. Dunwell, T. Lim, N. Suttie, S. Louchart, M. Hendrix and S. de Freitas. (2013, Nov.). The Development Approach of a Pedagogically-Driven Serious Game to support Relationship and Sex Education (RSE) within a classroom setting. Computers & Education, 69, pp. 15-30.

[16] P.D. da Silva-Simones and C.G.I. Ferreira, "Military war games edutainment," in Serious Games and Applications for Health (SeGAH), 2011 IEEE 1st International Conference on, Braga, 2011, pp. 1-7.

[17] C. García-García, J.L. Fernández-Robles, V. Larios-Rosillo and H. Luga. (2012). ALFIL: A Crowd Simulation Serious Game for Massive Evacuation Training and Awareness. International Journal of Game-Based Learning, 2(3), pp. 71-86.

[18] P. Backlund, H. Engstrom, C. Hammar, M. Johannesson and M.

Lebram, "Sidh - a Game Based Firefighter Training Simulation," in Information Visualization, 2007. IV '07. 11th International Conference, Zürich, 2007, pp. 899-907.

[19] F. Torres, F.A. Candelas, S. Puente, J. Pomares, P. Gil and F.G. Ortiz. (2006). Experiences with Virtual Environment and Remote Laboratory for Teaching and Learning Robotics at the University of Alicante. International Journal of Engineering Education, 22(4), pp. 766-776.

[20] F.A Candelas, S. Puente, F. Torres, F.G. Ortiz, P. Gil and J. Pomares. (2013). A Virtual Laboratory for Teaching Robotics. International Journal of Engineering Education, 19(3), pp. 363-370.

[21] C.A. Jara, F.A. Candelas, S. Puente and F. Torres. (2011, Dec.). Hands-on experience of undergraduate students in automatic and robotics using a virtual lab and remote laboratory. Computers & Education, 57(4), pp. 2451-2461.

[22] T.J. Mateo-Sanguinp and J.M. Andújar-Marquez. (2012, Dec). Simulation tool for teaching and learning 3D kinematics workspaces of serial robotic arms with up to 5-DOF. Computer Applications in Engineering Education, 20(4), pp. 750-761.

[23] E. Rohmer, S.P.N Singh and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, Tokyo, 2013, pp. 1321-1326.

[24] A.R. Sartorius-Castellanos, L. Hernández, E. Rubio and I. Santana. (2006). Virtual and Remote Laboratory for Robot Manipulator Control Study. International Journal of Engineering Education, 22(4), pp. 702-710.

[25] I. Santana, M. Ferre, E. Izaguirre, R. Aracil and L. Hernández. (2012, Jan). Remote Laboratories for Education and research purposes in Automatic Control Systems. Industrial Informatics, IEEE Transactions on, 9(1), pp. 547-556.

[26] M. Kaluz, L. Cirka, R. Valo and M. Fikar, "ArPi Lab: A Low-Cost Remote Laboratory for Control Education," in Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, 2014, pp. 9057-9062.

[27] H.H. Hahn and M.W. Spong, "Remote laboratories for control education," in Decision and Control, 2000, Proceedings of the 39th IEEE Conference on, Sydney, NSW, 2000, pp. 895-900.

[28] L. Ljung and T. Glad, "Modeling of Dynamic systems," PTR Prentice Hall, 1994.

[29] W. J. Palm III, "Modeling, Análisis and Control of Dynamic Systems," 2nd ed., John Wiley Sons, Inc.

[30] R. Isermann, J. Schaffnit and S. Sinsel. (1999, May.). Hardware-in-the-loop simulation for the design and testing of engine-control systems. Control Engineering Practice, 7(5), pp. 643-653.

[31] F. Pierrot, C. Reynaud and A. Fournier. (1990, Apr.).DELTA: a simple and efficient parallel robot. Robotica, 8(2), pp. 105-109.

Jorge-Alberto Ortega-Moody, obtuvo el grado de Ingeniero en Sistemas Electrónicos en el ITESM, México; el grado de Maestro en Ciencias en Sistemas de Manufactura en el ITESM, México; el Doctorado en el Instituto Politécnico Nacional, México; y Postdoctorado en la Universidad de Houston. USA. Actualmente es profesor tiempo completo del Instituto Tecnológico Superior de San Andrés Tuxtla, México. Trabaja con líneas de investigación en Diseño Mecatrónico, Control, Automatización, Realidad Virtual y Motores de Video Juegos aplicados a la Mecatrónica.

Róger E. Sánchez-Alonso. obtuvo el grado de Ingeniero Industrial en la Universidad Nacional de Ingeniería, Nicaragua, y el grado de Maestro en Tecnología Avanzada en el Instituto Politécnico Nacional, México, donde actualmente desarrolla estudios para obtener el grado de Doctor. Es profesor titular de la Universidad Nacional de Ingeniería, Nicaragua. Trabaja en líneas de investigación vinculadas con el Diseño y Análisis de Robots Paralelos, Procesamiento de Imágenes y Realidad Virtual.

José-Joel González-Barbosa, nació en Guanajuato, México, en 1974. Recibió el título de Ingeniería en Electrónica por parte de la Universidad de Guanajuato, México. Realizó su maestría y doctorado en Informática y Telecomunicaciones en el Institut National Polytechnique de Toulouse en 1998 y 2004 respectivamente. La estancia de investigación de la maestría y doctorado la realizó en el LAAS-CNRS de Toulouse. Profesor Investigador del CICATA Querétaro-IPN, México, donde imparte cursos de Visión por Computadora, Procesamiento de Imágenes y Clasificación de patrones. Sus áreas de intereses incluyen percepción y robótica móvil.

Guillermo Reyes-Morales, recibió el grado de Ingeniero Mecánico Electricista en la Universidad Veracruzana, México; el grado de Maestro en Ingeniería Electrónica en el Instituto Tecnológico de Orizaba, México. Actualmente cursa el Doctorado en Manufactura Avanzada en el CIATEQ, México. Trabaja en las líneas de investigación de Automatización y Control.